Speed-dependent Automatic Zooming

Josh Savage
Computer Science Department
Canterbury University
Christchurch, New Zealand
jps42@cosc.canterbury.ac.nz

Supervisor: Andy Cockburn

Abstract

Today's computer environment requires frequent browsing of documents too large for the viewing window. The current scroll and zoom interfaces are slow and inefficient. We present a new navigation technique called speed-dependent automatic zooming (SDAZ). It couples rate-based scrolling with automatic zooming. Zooming occurs automatically as the scroll speed increases, keeping the visual flow across the screen constant. This allows users to efficiently scroll a document without having to manually switch between zooming and scrolling or becoming disoriented by fast visual flow. We have designed two SDAZ interfaces — a map browser and a document browser. An evaluation of these interfaces against traditional interfaces showed that SDAZ is significantly faster and preferred by participants.

Contents

1 Introduction **5**

2 Related Work **7**
2.1 Alternative Scrolling Techniques . . . 7
2.2 Zooming Techniques . . . 8
2.3 Visualisation Techniques . . . 8
2.4 Speed-dependent Automatic Zooming . . . 8
2.4.1 Web Browser . . . 8
2.4.2 Map Browser . . . 8
2.4.3 Sound Editor . . . 9
2.4.4 Image Viewer . . . 10
2.4.5 Dictionary Viewer . . . 10
2.5 3-D Navigation Techniques . . . 11

3 Design and Implementation of Speed-dependent Automatic Zooming **13**
3.1 Document Browser . . . 14
3.2 Map Browser . . . 16

4 Evaluation **19**
4.1 Procedure . . . 19
4.1.1 Document Browsing . . . 20
4.1.2 Map Browsing . . . 21
4.2 Subjective Measures . . . 21
4.3 Experiment Concerns . . . 22

5 Results **23**
5.1 Document Browser . . . 23
5.1.1 Task Completion Times . . . 23
5.1.2 Subjective NASA-TLX Responses . . . 23
5.2 Map Browser . . . 24
5.2.1 Task Completion Times . . . 24
5.2.2 Subjective NASA-TLX Responses . . . 24
5.3 Observations and Comments . . . 25

6 Discussion **27**
6.1 Further Work . . . 28

7 Conclusion **29**

Chapter 1

Introduction

Document browsing is one of the most frequent tasks users face when interacting with computers. A document, such as a text file, a spreadsheet, a World Wide Web (WWW) page, or a map is often larger than the viewing window. Users are therefore faced with navigation problems in the browser. Most browsers facilitate document navigation primarily with scrolling (or panning) and zooming interfaces. Scrolling allows the user to move to different locations in the document, while zooming allows the user to view a target at different scales. However, both of these techniques have fundamental limitations.

The common problem with scrolling and zooming interfaces is that when users are zoomed out for orientation, there is not enough detail to do any 'real work'. When they are zoomed in sufficiently to see detail, the context is lost. To reduce this problem, multiple windows can be provided, each with pan and zoom capability. Although this is reasonable for small information spaces, the many windows required by large spaces often lead to usability problems due to excessive screen clutter and window overlap. An alternative strategy is to have one window containing a small overview, while a second window shows a large more detailed view (Beard & Walker 1990). The small overview contains a rectangle that can be moved and resized, and its contents are shown at a larger scale in the large view. This strategy, however, requires extra space for the overview and forces the viewer to mentally integrate the detail and context views. An operational overhead is also required, because the user must regularly move the mouse between the detail and context windows.

Common zooming interfaces require a user to move the mouse cursor back and forth between the document and the zoom buttons. Some zooming interfaces have a zoom interaction mode that allows a user to zoom in and out by clicking the left and right mouse buttons, respectively. This can reduce operation time depending on the type of task. For example, this feature is effective for tasks requiring specific zooming into a small location on a map. However, the user is still required to select the zoom mode from the side bar or menu, when (s)he wants to begin zooming and when (s)he wants to finish. These additional interaction modes can cause extra confusion for the user. For example, the user can forget which mode they are currently in and try to perform another operation, without switching modes.

Traditional scrolling interfaces suffer from two major problems. While using the scrolling interface the user must constantly move the mouse back and forth between the document and the scroll bar. This causes a significant increase in operation time. The other limitation becomes apparent when users scroll through very large documents. The scroll handle becomes very small and the slightest movement can cause a sudden jump to a different location in the document. This sudden shift can leave the user feeling disoriented and confused (Igarashi & Hinckley 2000).

An alternative approach is a rate-based scrolling interface (Zhai et al. 1993). Rate-based scrolling maps displacement of the input device to the velocity of scrolling. Two examples of hardware that uses rate-based scrolling are the Microsoft IntelliMouse™ and the IBM ScrollPoint II™. The Microsoft IntelliMouse™ provides a wheel for scrolling but can also map the mouse position to velocity. The IBM ScrollPoint II™ mouse maps the force exerted on a small joystick to velocity (Barrett et al. 1995). An advantage of these devices is that the user does not have to move the mouse cursor to the scroll bar. The disadvantage is that there is an upper limit for usable scrolling speed, because extremely fast scrolling causes disorientation. This means the user has to scroll slowly to a distant location.

Speed-dependent automatic zooming (SDAZ) is a navigation technique first proposed by Igarashi & Hinckley (2000). It uses rate-based scrolling to overcome the limitations of typical scrolling interfaces. The system couples this with automatic zooming to prevent extreme visual flow. This means that as a user scrolls faster the system automatically zooms out, providing a constant information flow across the screen.

Igarashi & Hinckley performed informal preliminary usability studies on two main SDAZ applications: a map browser and a web browser. This paper extends their work with an in-depth usability study with performance-improved SDAZ prototypes. Each prototype was assessed against a standard scroll and zoom interface to attempt to establish which one is more efficient. In addition, studies were conducted to identify any usability problems and assess subject preference.

Prior work on related systems is described in Chapter 2. The speed-dependent automatic zooming concept and new prototypes are described in detail in Chapter 3. Chapter 4 describes the evaluation of each of the new prototypes against traditional scrolling and zooming interfaces. Results, a discussion and conclusions then follow in Chapters 5, 6 and 7.

Chapter 2

Related Work

2.1 Alternative Scrolling Techniques

Several alterations to traditional scroll bars have been proposed. The AlphaSlider (Ahlberg & Shneiderman 1994) is an alternative scrolling technique for precise selection in large lists (see Figure 2.1(a)). The essential components of the AlphaSlider are the same as those of a drop-down list-box. There is a slider area, a slider thumb (the square in the middle that a user can directly drag on), and left/right arrow keys. The AlphaSlider functions identically to a drop-down list-box, except for its vertical orientation. The main advantage of the AlphaSlider interface is that it uses only one line of text output — it is very efficient in its screen space use. One of its disadvantages is that it allows less use of context in searching. With a typical drop-down list-box interface, there are other words in the interface from which the subject can see where (s)he is in the list and see how fast the alphabet is being moved through.

Two designs that improve upon the AlphaSlider are the FineSlider (Masui et al. 1995) and the Stretch Button Scrollbar (Smith & Henning 1996). The FineSlider is based upon the AlphaSlider but uses an elastic technique based on a rubber band metaphor (see Figure 2.1(b)). A control object such as a scroll bar is moved by pulling the object with a rubber band between the object and the mouse cursor. The Stretch Button Scrollbar uses a completely different approach (Figure 2.1(c)). The scroll handle is modified with two extra buttons, one on either side. These buttons allow fine movement similar to that provided by the arrow movement buttons on traditional scroll bars.

(a) The AlphaSlider. (b) The FineSlider. (c) The Stretch Button Scrollbar.

Figure 2.1: Alternative Scrolling Techniques.

2.2 Zooming Techniques

Studies have been carried out by Combs & Bederson (1999) to determine whether zooming improves image browsing. They found that zoomable user interfaces and 2-D interfaces perform equally, and both performed better than 3-D systems.

Pad (Perlin & Fox 1993) and Pad++ (Bederson & Hollan 1994) are zoomable user interfaces that provide continuous zooming as a central navigation tool. Objects are spatially organised in 2-D space, and the user acquires an object by using scrolling and zooming operations. The drawback to this type of interface is that a user must control both scrolling and zooming interfaces. Speed-dependent Automatic Zooming (Igarashi & Hinckley 2000) attempts to integrate scrolling with zooming so that users need only be concerned with required information. Neither, Pad or Pad++ have been empirically evaluated.

2.3 Visualisation Techniques

Information visualisation techniques, such as Fisheye (Furnas 1986, Gutwin 2002), Holophrasting (Cockburn 2001, Smith & Cockburn 2002), Document Lens (Perlin & Fox 1993), and Perspective Wall (Perlin & Fox 1993) are techniques used to reduce information overload. They reduce the size of a document by magnifying the focused area, while the non-focused areas are compressed but still remain visible. Unfortunately, this can cause several problems for users. One of these is focus-targeting, where a user moves the focus to a new location. Magnification makes focus-targeting difficult because objects appear to move as the focus point approaches them. One solution to this problem is speed-coupled flattening (Gutwin 2002). This technique reduces the distortion level of the focus, based on pointer velocity and acceleration. Speed-coupled flattening was proven to significantly reduce both targeting time and targeting errors in fisheye environments.

2.4 Speed-dependent Automatic Zooming

The concept of speed-dependent automatic zooming (SDAZ) was first proposed by Igarashi & Hinckley. They implemented several SDAZ prototype systems in the Java™ programming language: a web browser, a map viewer, a sound editor, an image browser, and a dictionary viewer. Of these, the web browser and map viewer were successful in providing a potentially more efficient browser. These were used to perform preliminary usability studies to try and clarify the strengths and limitations of automatic zooming. The following sub sections describe their implementations and results for each browser.

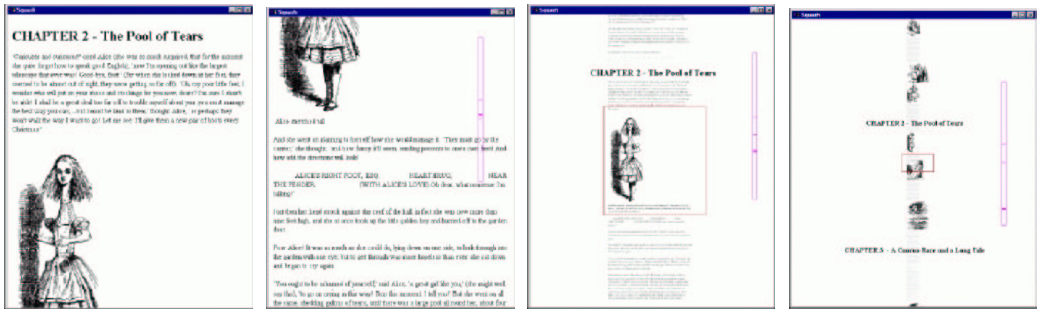
2.4.1 Web Browser

Igarashi & Hinckley's prototype web browser implemented in Java™ is shown in Figure 2.2. When the user presses the mouse button, a pink slider showing the document scroll speed appears on the right hand side of the screen. The scroll speed is increased by moving the mouse up or down while holding down the button. As the scroll speed increases, the document text becomes smaller, giving an overview of the document. When the user releases the mouse button, the text gradually returns to original size.

The web browser study timed seven subjects completing one task. The task consisted of finding different images in different places in a web document. They found that subjects using SDAZ and conventional scrolling performed approximately equally well, even though subjects had little experience with the new SDAZ interface. They also found that six out of seven subjects preferred automatic zooming, which suggests it might be a suitable alternative to traditional scrolling.

2.4.2 Map Browser

Igarashi & Hinckley's map browser (Figure 2.3) has two methods of interaction. One method is holding the mouse button down and dragging the mouse. The relative position between the point of the initial click of the mouse button and the current mouse position specifies the direction and speed of movement. The



(a) Static view. (b) Scrolling slowly. (c) Scrolling fast. (d) Scrolling very fast.

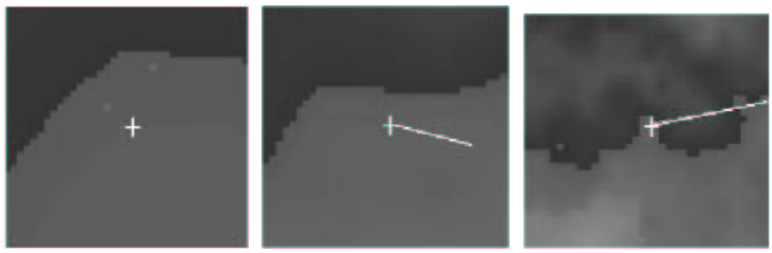
Figure 2.2: Igarashi and Hinckley's Prototype Web Browser.

second method of interaction is using a joystick. The more the user tilts the stick, the faster (s)he scrolls, and the smaller the view gets. This method was used in user evaluations.

The map browser uses an artificially generated map based on Perlin's noise function (Perlin 1985). This enabled Igarashi & Hinckley to test the idea of the map browser with minimum implementation effort, as well as high performance.

The evaluation consisted of users locating different targets in a 2-D map using traditional scrolling/zooming and SDAZ. The task completion time results were mixed compared to the web browser. The authors suggest that this was because the map navigation task was more difficult than browsing a document, which caused users to employ a range of different strategies. The most efficient strategy is to zoom out until the target appears on the screen, move to the target, and then zoom in. However, some subjects slowly moved to the target without zooming out, which took a very long time.

The results from the subjects' qualitative evaluations were also mixed. Four out of the seven subjects preferred the SDAZ interface.



(a) Static view. (b) Scrolling slowly. (c) Scrolling fast.

Figure 2.3: Igarashi and Hinckley's Prototype Map Browser.

2.4.3 Sound Editor

A sound editor with automatic zooming was implemented but found by Igarashi & Hinckley to be unusable. The continuously transforming waveform was too confusing. This, combined with the lack of appropriate visual landmarks needed for automatic zooming targeting, made the implementation very difficult to use.

2.4.4 Image Viewer

The prototype image browser was used to browse a collection of images (Figure 2.4). The images were aligned in a horizontal list and a user browsed by moving along the list. As the scrolling speed increased, the images become smaller so that more of them could be placed on the screen.

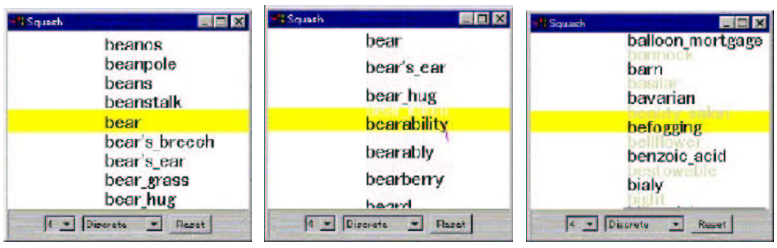
(a) Static view. (b) Scrolling slowly. (c) Scrolling fast. (d) Scrolling very fast.

Figure 2.4: Igarashi and Hinckley's Prototype Image Viewer.

Igarashi & Hinckley found that their automatic zooming image browser was less effective than a standard static array of thumbnails. This is because the image browser lacked the same abstraction given by the other browsers. For example, when browsing with the map browser, narrow streets fade away in the zoomed-out view, and highways and area labels become landmarks. In the web browser, the smaller body of text becomes blurry and only pictures and headings are clear for navigation. In the Image Browser, however, each image is unique and the order of the images is not important, so the same abstraction is missing.

2.4.5 Dictionary Viewer

The zooming-out effect in the dictionary viewer is achieved by pruning out the less important items. Figure 2.5 shows screen shots from an Igarashi & Hinckley paper, which show their dictionary viewer in execution. In this example, the words are displayed in alphabetical order and are scrolled vertically by holding the middle button down and moving the mouse up and down. As the user scrolls faster, the list starts to skip words.



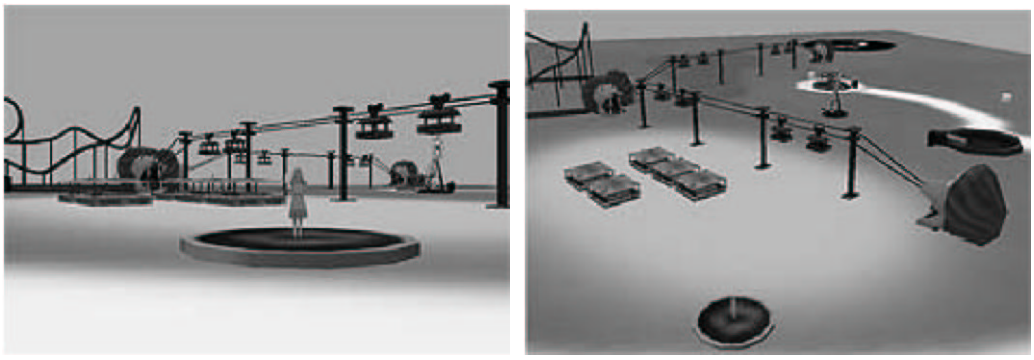
(a) Static view. (b) Scrolling slowly. (c) Scrolling fast.

Figure 2.5: Igarashi and Hinckley's Prototype Dictionary Viewer.

Igarashi & Hinckley found the dictionary viewer confusing to use because words appear and disappear during scrolling. In addition, they found it required a significant cognitive overhead to locate a target word in the zoomed-out view because the user has to constantly think about the alphabetical order between the visible words and the target word. The example given by Igarashi & Hinckley shows that when searching for "bear", the user has to steer between "bavarian" and "befogging" in the zoomed-out view.

2.5 3-D Navigation Techniques

Depth-modulated flying (Ware & Fleet 1997) and speed-coupled flying with orbiting (Tan et al. 2001) are both techniques used to navigate in a 3-D world. Depth-modulated flying uses continuous height sampling to control the flying speed. The idea being, to fly slower closer to the ground and faster in the higher overview. Speed-coupled flying with orbiting uses the opposite approach. It modulates the height and tilt of the camera related to the flying speed (Figure 2.6). SDAZ is essentially based on this navigation technique, but in 2-D space.



(a) Local view of scene while standing still or moving slowly.

(b) Overview of scene while moving fast.

Figure 2.6: Speed-coupled flying with orbiting.

Chapter 3

Design and Implementation of Speed-dependent Automatic Zooming

The concept of speed-dependent automatic zooming (SDAZ) is to adjust the zoom level automatically as the scroll speed changes, in order to prevent extreme visual flow during rate-based scrolling. Equation 3.1 shows the relationship between zoom level and scroll speed. The zoom level indicates the perceived distance from the document. Therefore, if the zoom level is increased, the document is perceived to be further away and hence smaller. This is consistent with Igarashi & Hinckley's observation that users move slowly when focusing on details and quickly when focusing in the global overview.

$$zoomlevel = k \times scrollspeed - threshold \tag{3.1}$$

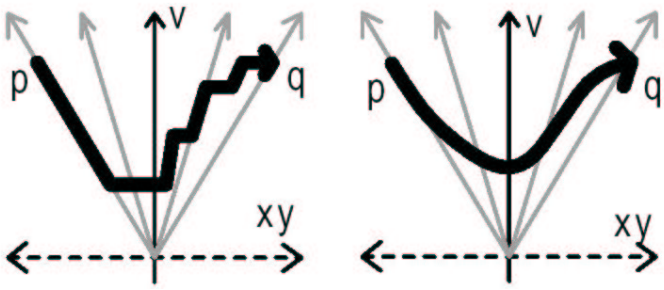
The threshold constant is used to ensure that the perceived scrolling speed on the screen (the visual flow of the document across the screen) remains constant regardless of the actual scrolling speed in the information space. If the zoom level becomes negative because of the threshold it is rounded to zero. An increase in the threshold requires a greater document scroll speed before zooming out begins. The constant k respresents the speed at which the zoom level is increased. A larger k value will increase the zooming speed of the document.

We have three specific goals for the use of the SDAZ technique. Our main goal is to allow the user to target a position quickly without becoming annoyed or disoriented by extreme visual flow. Second, we want the technique to provide smooth transitions between the magnified local view and the global overview. Our final goal is to intergate document zooming with scrolling, so that the user can zoom efficiently without having to manually change the document magnification factor.

The efficiency of navigation in a document using SDAZ can be explained using Igarashi & Hinckley's space-scale diagram (Furnas & Bederson 1995) (See Figure 3.1). In traditional manual zooming interfaces, the user must manually control zooming and scrolling, resulting in the zigzag pan-zoom trajectory shown in Figure 3.1(a). Automatic zooming, however, is more efficient because it has a smooth curve-shaped trajectory, due to the zoom level changing automatically with the scroll speed (See Figure 3.1(b)).

A SDAZ web-browser interface was initially implemented using Microsoft Visual Basic. The browser was designed so that as the scrolling speed increased, the size in the text and images of the document decreased. Text header sizes could be reduced at a slower rate than the text body. This gave an efficient focus plus context visualisation when zooming, because the heading sizes, were made clearly readable while the text body became small and blurry. The implementation was unsuccessful, however, because the browser became extremely jerky towards the end of the text document. This was caused by the browser updating its scroll position each time the text size changed, because the document size also changed with the text.

The improved prototypes (a map browser and a document browser) were designed in the programming language C using the OpenGL library packages. The OpenGL library was used as it provides high performance 2-D and 3-D graphics suitable for creating smooth document and map browsers. Sections 3.1 and 3.2 describe in detail the design and implementation of each of the prototypes.



(a) Manual zooming/scrolling (b) Automatic zooming

Figure 3.1: Space-scale diagram of the pan-zoom trajectory. (v=scale, xy=space, p=initial position, q=target position)

To enhance the smoothness of both browsers' transition between the magnified local view and the global overview a 'falling' speed was introduced. This technique avoids the user having the document suddenly magnified when the mouse button is released; rather it is smoothly brought back to the magnified view at the set falling rate. The falling rate was calculated using trial and error. We found that if the rate was too fast, the user felt motion sickness and lost placement in the document, whereas if it was too small, the user became frustrated because (s)he was forced to wait long periods of time before the document was fully remagnified.

3.1 Document Browser

The document viewer was designed to browse portable document format (PDF) and postscript (PS) files using automatic zooming. The PDF or PS file was first converted into smaller Truevision Targa (TGA) files (one TGA for each page). This conversion would be unwise in a commercial version of the browser, as it is initially time consuming. In our prototype it served two purposes. First, TGA files are easier to read into C and texture map in OpenGL. Secondly, by splitting a large document into smaller images, OpenGL does not need to render the unseen pages and less video memory is used. This increases the speed and smoothness of the browser.

The implementation of the browser was simple but very efficient. It texture mapped the document pages to basic OpenGL rectangles. Because OpenGL renders objects in 3D, automatic zooming of a document is achieved very simply by moving the view point away from the rectangle. Equations 3.2 and 3.3 show the formula used to calculate the relationship between the mouse movement and the distance (or zoom level) from the document.

$$scrollspeed = |Y_{ip} - Y_{cp}| \tag{3.2}$$

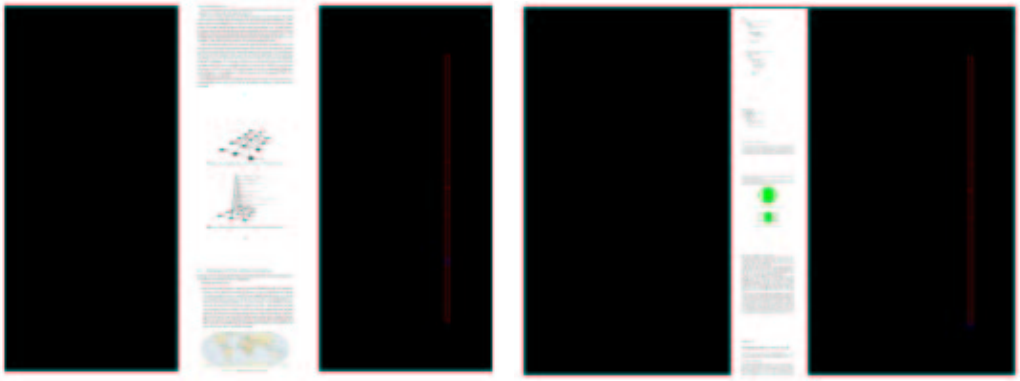
$$distance = k \times scrollspeed - threshold \tag{3.3}$$

(The distance variable indicates the distance in openGL the viewer is moved away from the document. k and threshold are constants representing the zoom speed and threshold before zooming begins, respectively. The scroll speed variable is the rate at which the document is scrolling. Both Y_{ip} and Y_{cp} represent the mouse Y co-ordinates received from the mouse while the left button is held down. Y_{ip} is the first recorded Y value when the mouse button is held down. Y_{cp} is the current Y position value of the mouse.)

Figure 3.2 shows screen shots of the prototype document browser. When the user holds down the mouse button a red box representing a scroll bar is displayed on the right side of the screen (See Figure 3.2(a)). The blue line inside the scroll bar is directly related to the scroll speed and movement direction of the document. The blue line starts in the middle of the red box and is moved up or down by moving the mouse

(a) Static

(b) Slow Scroll Speed



(c) Fast Scroll Speed

(d) Fastest Scroll Speed

Figure 3.2: Prototype Document Browser Screen Shots

in the same direction, while the mouse button is held down. For example, if the user wishes to scroll up, (s)he moves the mouse up while holding the mouse button down. The dotted red lines show the automatic zooming threshold. If the measured level of scroll speed exceeds these lines the document automatically begins to zoom out, shown in Figures 3.2(c) and 3.2(d). After the user has found the required target, (s)he can release the button, and the document is smoothly remagnified from the centre. Figure 3.2(d) shows the global overview achieved when the scroll speed is at its maximum.

One of the problems associated with coupling scroll speed to zoom level arises when the user rapidly increases or decreases the scrolling speed. For example, when changing direction from moving down through the document to up, the user drags the mouse up, rapidly reducing scroll speed in the current direction, and then the scroll speed is increased in the opposite direction. These sudden changes in scroll speed cause a rough transition between the magnified view and zoomed view. To address this problem, we used the falling rate (see Section 3) to limit the remagnification of the document. Thus, the document can only be remagnified at the set falling speed regardless of the current scroll speed. We found in our initial experiments however, that users felt less in control, as they could not remagnify the document as fast as they liked. In the performance evaluations for the document browser, we did not use this feature.

3.2 Map Browser

The implementation of the map browser was similar to that of the document browser, expect the map browser browses large maps and allows two dimensional movement (in both X and Y directions). A map image to be rendered with the prototype map browser must be divided into six equal sections, each saved in the TGA image format. The prototype pieces the sections together to form the map. By dividing the map into separate images, it can be displayed with greater detail, because openGL has a maximum image size limit of 2048 x 2048 pixels. Thus, the more images the map is divided into, the greater the detail of the map. We found that six images gave the prototype the required detail for locating targets in our road map.

Equations 3.4 and 3.5 were used to calculate the scrolling speed and the zoom level (distance away) from the map. The zoom level is calculated in the same way as in the document browser. The scrolling speed, however, is directly related to the line between the cross, displayed on screen when the left mouse button is pressed and the current cursor position.

$$scrollspeed = \sqrt{(Y_{ip} - Y_{cp})^2 + (X_{ip} - X_{cp})^2} \tag{3.4}$$

$$distance = k \times scrollspeed - threshold \tag{3.5}$$

To use the map browser, the user first holds down the left mouse button, which causes a red cross to be displayed in the centre of the screen (see Figure 3.3(a)). As the user moves the mouse with the button pressed, a vector appears between the mouse cursor and the red cross (see Figure 3.3(b)). This vector represents the speed and direction of the movement around the map. As the user increases the size of the vector, the zoom level automatically increases to allow a greater overview of the map (see Figures 3.3(c) and 3.3(d)). If the mouse button is released, the document is smoothly remagnified to the local view, as with the document browser.

The problem of rapid re-zooming associated with sudden changes in direction was also found to be an issue for map browsing. In addition, the problem significantly increased in the map browser, because it was more common for users to want to change direction while trying to locate a target (as compared to a textual document). This problem was solved using the falling rate (see Section 3) to limit the remagnification rate. Thus, the map could only be remagnified at the set falling speed regardless of the current scroll speed. We found this technique to be very natural for users when browsing maps, as it simulates gravity.

(a) Static

(b) Slow Scroll Speed

(c) Fast Scroll Speed

(d) Fastest Scroll Speed

Figure 3.3: Prototype Map Browser Screen Shots

Chapter 4

Evaluation

The goals of the evaluation were to compare the efficiency of the new automatic zooming technique to a standard scroll and zoom technique for document and map browsing. In addition, we wanted to identify any usability problems and to assess subjective preferences. The 12 volunteer participants (11 males and 1 female) were all computer science graduate students.

4.1 Procedure

The evaluation consisted of measuring the time required to locate different targets using our two prototype automatic zooming interfaces and two traditional manual scroll and zoom interfaces. Acrobat Reader 5 and Paint Shop Pro 5 were used as the traditional document and map browsing systems, respectively, because they contain the traditional methods of navigation and are commonly used for browsing such documents.

The Acrobat Reader interface is shown in Figure 4.1(a). To zoom a user must first select the zooming mode (zoom in or zoom out from the document). This is done by selecting the plus or minus magnifying glass from the zoom menu (See Figure 4.1(b)). Zooming is then performed by clicking on the document with the left mouse button. Alternatively, a user in zooming mode can then right click on the document and select the appropriate zoom level from the pop up menu (See Figure 4.1(c)). After a zoom level is selected, the document is instantly zoomed to that level. If the document is being remagnified, the page at the top of the screen always remains at the top. When remagnifying in the SDAZ interfaces, the centre is always smoothly bought back into focus.

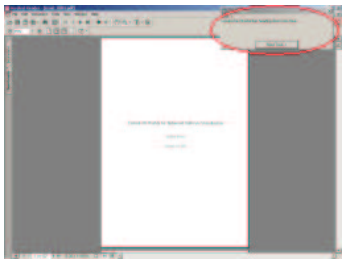
Figure 4.2(a) shows screenshots of tasks being completed using Paint Shop Pro 5. The user zooms in by clicking the left mouse button on the area of the image to be magnified and zooms out using the right button. When an area is clicked, it moves to the centre of the screen. This is shown in Figures 4.2(b) and 4.2(c). When the user clicks in the Massey Campus area in the top left corner of the window, it is magnified and moved to the centre of the window (Figure 4.2(c)). A user scrolls using the traditional scroll bars found at the left and bottom sides of the map.

At the beginning, of the evaluation participants were asked to complete a questionnaire inquiring about how regularly they play computer games and if they had previously used Paint Shop Pro 5 or Acrobat Reader 5. The 12 participants past experience is summarised in Table 4.1.

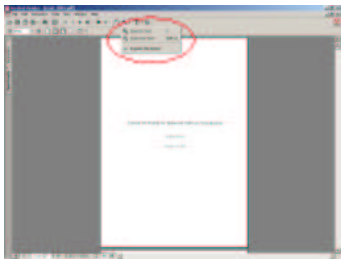
Immediately prior to being tested with each interface, participants were allowed five minutes to become familiar with that interface. Participants were given two sample tasks; one sample task for each task type.

Game Play?					**Have used**	**Have used**
Never	1-2 hrs pw	2-3 hrs pw	5-10 hrs pw	20+ hrs pw	Paint Shop Pro 5?	Acrobat Reader 5?
3	1	2	4	2	6	12

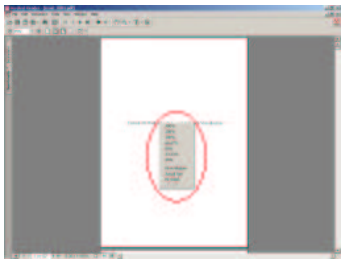
Table 4.1: Summarised particpants' experience.



(a) Tasks being completed in Acrobat Reader with instructions in the top right hand corner



(b) Screenshot showing the location of the zoom menu bar.



(c) Screenshot showing the right button zoom menu. Options include magnifying the document to 400%, 200%, 100%, 66.67%, 50%, 33.33%, 25%, to fit in window, to the actual size, and to fit to the document width.

Figure 4.1: Screenshots taken of the Acrobat Reader 5 Interface.

(a) Tasks being completed in Paint Shop Pro with instructions in the top right hand corner

(b) Map view before mouse clicked on Massey Campus.

(c) Map view magnified on Massey Campus.

Figure 4.2: Screenshots taken of the Paint Shop Pro 5 Interface.

4.1.1 Document Browsing

The document browsing evaluations timed participants performing eight tasks varied by distance and task type for each of the interfaces (SDAZ document browser and Acrobat Reader 5). There were two distances, short (approximately 5 pages to the target) and long (approximately 20 pages to the target). The document browsing task types were as follows:

- Locate Picture Task — This required the subject to locate a picture in the document given a description and direction relative to their current position. A typical locate picture task would read “Locate the picture of the world globe up from here”.
- Locate Text Heading Task — This required the subject to locate a text heading in the document given the heading title and the direction. A typical locate text heading task would read “Locate the Software Visualization heading down from here”.

Each subject performed one task twice for each distance with each task type; thus (s)he needed to locate eight schools in total for each interface.

The tasks were timed using a stopwatch controlled by the evaluator. The task instructions were displayed in the top right hand corner of the screen in both interfaces thoughout the evaluation (see Figure

4.1(a)). Before each task, the evaluator read the instructions to the subject. When the subject was ready to begin, the evaluator counted down from three and then the stopwatch was started. Throughout the task, the subject was allowed to ask the evaluator to read the instructions again or to point in the direction of the start point. The stopwatch was stopped when the first word below the target was read aloud. For example, if the subject was searching for a text heading, (s)he would read the first word of the text body below the heading. For a picture, (s)he would read the first word of the caption below the picture.

4.1.2 Map Browsing

The map browsing evaluations timed participants using the two interfaces (the prototype SDAZ map browser and Paint Shop Pro 5) to locate schools (marked in yellow) on an Auckland road map. The tasks were varied by distance to the target and the task type. The distances used were short (approximately 5 grid squares to the target) and long (approximately 20 grid squares to the target). The two map browsing task types were as follows:

- Locate Task — This required the subject to locate a school when given the suburb and a direction from the last school. For example, a typical task would read "Locate Wairau Intermediate School in SunnyNook, northwest of here".
- Follow Task — This required the subject to follow one or two state highways until they found the given school. For example, a typical task would read "Follow state highway 16 north; when you reach state highway 1, follow this until the Auckland Institute of Technology".

Each subject performed one task twice for each distance with each task type; thus (s)he needed to locate eight schools in total for each interface.

The tasks were timed using a stopwatch controlled by the evaluator. The task instructions were displayed in the top right hand corner of the screen in both interfaces throughout the evaluation (see Figure 4.2(a)). Before each task, the evaluator read the instructions to the subject. When the subject was ready to begin, the evaluator counted down from three and then the stopwatch was started. Throughout the task, the subject was allowed to ask the evaluator to read the instructions again or to point in the direction of the start point. After the subject located the school and roughly centred it on the screen the stopwatch was stopped.

4.2 Subjective Measures

To gather subjective responses to the workload required in the new interfaces, a standard NASA Task Load Index (TLX) worksheet (Hart & Staveland 1988) was used. The six categories explained as follows were recorded on a five-point Likert-scale from low to high (for performance, the scale endpoints were labelled good to poor), with low being best and high being worst (for performance, good was best and poor was worst).

1. Mental Demand — "How much mental and perceptual activity was required (e.g. thinking, deciding, calculating, remembering, looking, searching, etc.)? Was the task easy or demanding, simple or complex, exacting or forgiving?"
2. Physical Demand — "How much physical activity was required (e.g. pushing, pulling, turning, controlling, activating, etc.)? Was the task easy or demanding, slow or brisk, slack or strenuous, restful or laborious?"
3. Temporal Demand — "How much time pressure was felt due to the rate or pace at which the tasks or task elements occurred? Was the pace slow and leisurely or rapid and frantic?"
4. Effort — "How hard did you have to work (mentally and physically) to accomplish your level of performance?"
5. Performance — "How well were the tasks completed? How satisfied were you with your performance in accomplishing these goals?"

6. Frustration Level — “How discouraged, irritated, stressed, and annoyed versus gratified, content, relaxed, and complacent did you feel during the tasks?”

After each interface was tested, the participants were asked to record a response for each category in the NASA-TLX worksheet. Participants were also asked to respond to the questions “What did you like about this interface?” and “What did you dislike about this interface?”. At the conclusion of the evaluation participants were asked which interfaces they preferred overall for the given tasks.

4.3 Experiment Concerns

A number of potential outside factors in the evaluations were identified and controlled.

- *Learning effects.* The order of the interfaces was varied among users. Six participants used the SDAZ interfaces first, and six participants used the traditional interfaces first. The sets of eight tasks were also varied equally among interfaces. Subjects always performed document browsing first, then map browsing in each interface type, in an attempt to prevent learning effects of the material viewed. Users were allowed to practice with each interface immediately prior to being tested until they became familiar with that interface.

- *Machine performance.* All tests were performed on the same machine: a 1.2GHz AMD Athlon with 640Mb of RAM running Windows XP, with a Geforce 4 MX video card outputting a resolution of 1024 x 764 pixels to a 19-inch IBM monitor. Whenever an evaluation was performed, the number of active processes was reduced to a minimum in order to ensure maximum performance of the CPU and video card. Input was given through a standard 3-button Logitech mouse, which was cleaned after each evaluation.

- *Bias through subject hand-eye co-ordination.* All evaluations were performed within subjects. If users had particularly good hand-eye co-ordination, this characteristic would be evident in both the SDAZ and traditional interface timing results.

- *Bias through inconsistent instructions.* The same task instructions were displayed on the screen for each participant. They were also read aloud by the evaluator for clarification.

Chapter 5

Results

5.1 Document Browser

5.1.1 Task Completion Times

The task completion time data for the document browsing tasks was analysed using a 2 x 2 x 2 repeated measures analysis of variance (ANOVA). The three factors (all within subjects) were:

- Interface Type: SDAZ and traditional scrolling and zooming.
- Task Type: Locate text heading and locate picture.
- Distance: Small (approximately 5 pages) and large (approximately 20 pages).

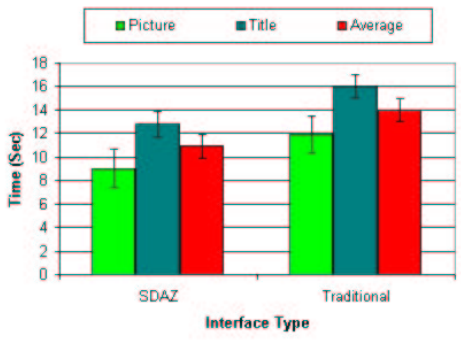
The fastest interface overall was SDAZ, with a mean acquisition time of 10.91 seconds and a standard deviation of 6.53 seconds. Acquisition time was 22% quicker with SDAZ than with the traditional scroll and zoom interface (Acrobat Reader). The traditional interface had a mean acquisition time of 13.98 seconds and a standard deviation of 6.95 seconds. There was a significant main effect between the two interface types ($f_{1,11} = 16.906, p < .002$).

As expected, there were significant main effects for factors distance ($f_{1,11} = 35.333, p < .001$) and task type ($f_{1,11} = 12.262, p < .005$). These are shown in Figure 5.1. The mean target acquisition times increased between small and large distances and between locate picture and locate title tasks.

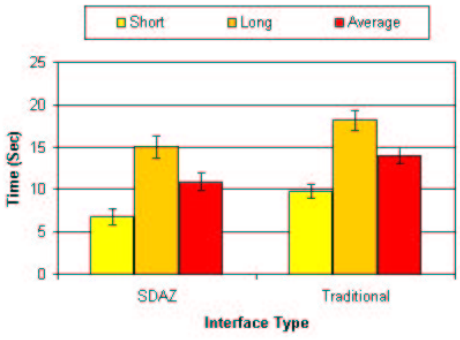
No significant interaction between interface type and task type was found. This is demonstrated by the relatively consistent increase in time between locate picture and locate title tasks, for both interface types (Figure 5.1(a)). There was also no significant interaction between interface type and distance. This is shown in Figure 5.1(b), again demonstrated by the relatively consistent increase in time as the distance increases, for both interface types.

5.1.2 Subjective NASA-TLX Responses

The means and standard deviations for the NASA-TLX worksheet are summarised in Table 5.1 and Figure 5.3. The Wilcoxon Matched-Pairs Test was used to prove a significant difference ($p < .05$) between the ratings for each interface. The z and p values for the test are shown beside the means and standard deviations in Table 5.1. SDAZ was preferred over traditional, in all categories. All categories were significantly different, excepting Mental Demand.



(a) Document task completion times by task type. The right-hand bar for each interface shows the overall mean for the interface across the two task types.



(b) Document task completion by distance.

Figure 5.1: Mean document task completion times for the two interfaces across task type and distance. Error bars show one standard error above and below the mean.

5.2 Map Browser

5.2.1 Task Completion Times

The task completion time data for the map browsing tasks was analysed using a 2 x 2 x 2 repeated measures ANOVA. The three factors (all within subjects) were:

- Interface Type: SDAZ and traditional scrolling and zooming.
- Task Type: Locate task and follow task.
- Distance: Small (approximately 5 map grid squares) and large (approximately 20 map grid squares).

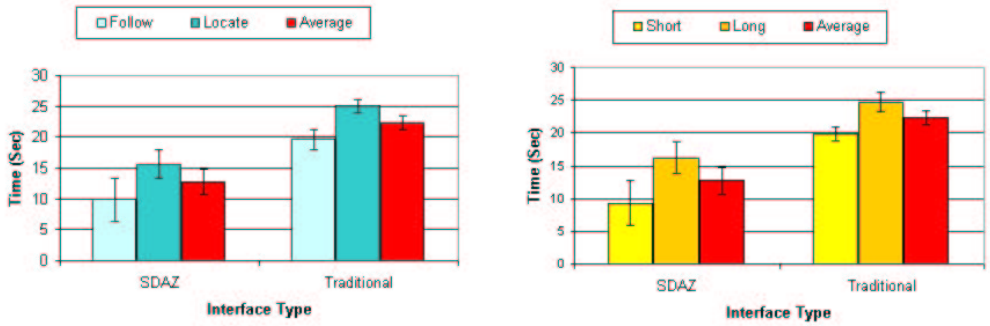
The fastest interface overall was SDAZ, with a mean acquisition time of 12.73 seconds and a standard deviation of 7.22 seconds. Acquisition time was 43% quicker with SDAZ than with the traditional scroll and zoom interface (Paint Shop Pro 5). The traditional interface had a mean acquisition time of 22.32 seconds and a standard deviation of 14.47 seconds. There was a significant main effect between the two interface types ($f_{1,11} = 38.866, p < .001$).

As expected, there were significant main effects for factors distance ($f_{1,11} = 5.504, p < .039$) and task type ($f_{1,11} = 5.632, p < .037$), illustrated in Figure 5.2. The mean target acquisition times increased between small and large distances and between locate and follow tasks.

No significant interaction between interface type and task type was found. This is demonstrated by the relatively consistent increase in time between follow and locate tasks, for both interface types (Figure 5.2(a)). There was also no significant interaction between interface type and distance. This is shown in Figure 5.2(b), and demonstrated again by the relatively consistent increase in time as the distance increases, for both interface types.

5.2.2 Subjective NASA-TLX Responses

The means and standard deviations for the NASA-TLX worksheet are summarised in Table 5.1 and Figure 5.3. The Wilcoxon Matched-Pairs Test was used to prove that all means were significantly different ($p < .05$). The z and p values for the test are shown beside the means and standard deviations in Table 5.1. SDAZ was preferred over traditional, in all categories.



(a) Map task completion times by task type. The right-hand bar for each interface shows the overall mean for the interface across the two task types.

(b) Map task completion by distance.

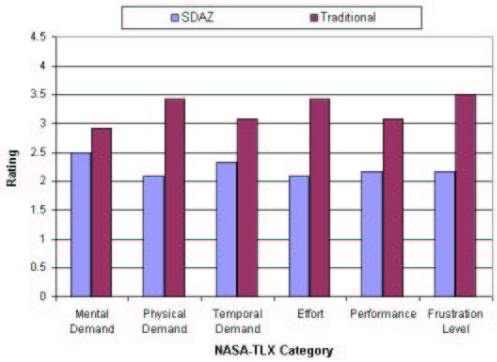
Figure 5.2: Mean map task completion times for the two interfaces across task type and distance. Error bars show one standard error above and below the mean.

5.3 Observations and Comments

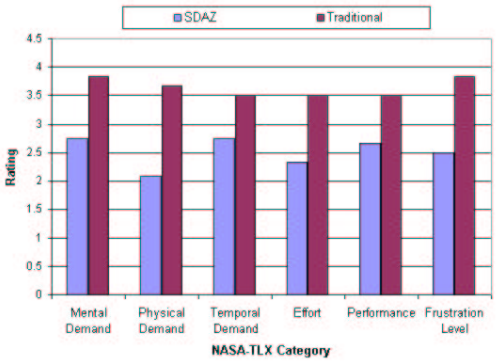
SDAZ for both document and map browsing was preferred by almost all participants. They felt that SDAZ was easy and natural to use because automatic zooming provided a smooth transition between views. Most participants reported that they felt more control in the SDAZ interfaces because they only needed to control one tool (the scroll speed). Only one participant preferred the traditional document browser, because he did not like being forced to scroll while zoomed out. This participant would have preferred some method to pause the zoom level in the SDAZ interface. The participant did, however, still prefer the SDAZ map browsing interface.

The traditional interfaces were rated less satisfactory primarily because participants felt the zooming and scrolling features were too separated. Participants could not zoom and scroll at the same time and were forced to frequently move the mouse cursor away from their work to the scroll bars. Participants also found the zooming very disorienting and confusing due to the abruptness of the transitions between zoom levels. In both Acrobat Reader and Paint Shop Pro, participants were constantly left wondering what their last point of focus was. This sometimes caused the participants to forget the task they were trying to complete before zooming because they were concentrating so hard on re-orienting themselves. We noticed that this was more common in the Paint Pro Shop interface because of its zoom centring property (See Figures 4.2(b) and 4.2(c)). In other words, the point that the display is to be zoomed in on is moved to the centre of the screen.

We observed that most participants typically set the zoom level only once in Acrobat Reader. This adjustment usually allowed one or two complete pages to be viewed in the window. Participants then used two main methods to navigate through the document with the scroll bars. One method involved clicking with the mouse on the scroll handle and dragging it until the target was found. Using the other method, participants moved the scroll handle by clicking between the handle and the arrow scroll buttons at the top and botton of the scroll bar. We noticed participants using this technique seemed to complete the tasks more quickly.



(a) Map Browsing Mean NASA-TLX Response Ratings.



(b) Doc Browsing Mean NASA-TLX Response Ratings.

Figure 5.3: Mean NASA-TLX Response Ratings.

Category	DOCUMENT BROWSING						MAP BROWSING					
	SDAZ		Traditional		Wilcoxon		SDAZ		Traditional		Wilcoxon	
	Mean	S.D.	Mean	S.D.	z	$p <$	Mean	S.D.	Mean	S.D.	z	$p <$
Mental	2.5	1.09	2.92	1.00	1.13	0.130	2.75	0.97	3.83	0.72	2.61	0.005
Physical	2.08	0.79	3.42	0.67	2.75	0.003	2.75	0.97	3.83	0.72	2.61	0.005
Temporal	2.33	0.65	3.08	1.08	1.84	0.033	2.75	0.66	3.50	0.91	1.90	0.029
Effort	2.08	0.52	3.42	0.79	2.75	0.003	2.33	0.65	3.50	0.80	2.75	0.003
Performance	2.17	0.58	3.08	0.52	2.45	0.007	2.67	0.49	3.50	1.17	2.09	0.018
Frustration	2.17	0.72	3.50	0.67	2.75	0.003	2.50	0.52	3.83	1.12	2.58	0.005

Table 5.1: NASA-TLX Ratings.

Chapter 6

Discussion

The results of the evaluation indicate that target acquisition in documents and maps is faster using SDAZ than traditional scroll and zoom techniques. This is because the zoom level is coupled with scroll speed, so that participants need to control only one method of navigation, leaving them free to concentrate on completing tasks as quickly as possible. In the traditional interfaces, especially with map browsing, participants frequently had to switch between the zoom level and the scroll position of the document. The combination of this factor and the jerky transition between the zoom levels caused frustration and cognitive overload in most subjects.

SDAZ was also preferred by all but one participant. That participant would have preferred SDAZ if there was a way to pause the re-zooming of the document. This feature can easily be associated with the right mouse button. When the right button is pressed, the user stops re-zooming but can still zoom out and scroll around the document. When the user wishes to re-zoom, (s)he clicks the right mouse button again.

The NASA-TLX responses indicate that users found the SDAZ interfaces:

- *Less mentally demanding* because automatic zooming requires less thought in regard to navigation, allowing users to focus more on the task itself.
- *Less physically demanding* because users did not have to constantly switch between the scroll and zoom tools.
- *Less temporally demanding* because they felt more in control with the SDAZ technique, and less pressured to complete the task.
- *Required less effort* for both mental and physical demands.
- *Provided greater performance*, because SDAZ required less effort to perform tasks, and participants felt they completed the tasks more efficiently.
- *Less frustrating* because users were able to complete the tasks easily and efficiently.

The only advantage users felt traditional scroll and zoom interfaces had over SDAZ interfaces was the ability to scroll to a known location in a document. For example, if the users knew that the information they were searching for was at the end of the document, they were forced to scroll the whole document before they could reach it (in SDAZ interfaces). To solve this problem, a traditional scroll bar could be added to the SDAZ interface, which would also enable users to see where they are in the document.

Igarashi & Hinckley's evaluation results differed significantly from ours. They found that SDAZ interfaces were on average the same as or less efficient than traditional interfaces for map and document browsing. The difference in results could be caused by several factors. For example, the smoothness of their prototype browsers could have influenced the participants' interactions with the interfaces. In addition, map data that we used was not artificially generated; real city road maps were used. Our evaluation used a range of different tasks, varying from finding text titles and pictures in the documents, to locating schools in maps with compass directions or a state highway to follow.

6.1 Further Work

The major disadvantage of SDAZ interfaces is the inability to scroll directly to a known location. Evaluations of mixed SDAZ and traditional scrolling interfaces need to be performed to establish whether this added functionality is more or less efficient for users. We believe that users will need to be specifically trained to use the scroll bar only when they know where the target is and to use SDAZ for all other searching tasks.

Further usability studies need to be conducted to highlight the strengths and weakness of SDAZ interfaces with advanced users. The advanced users need to be trained in both the SDAZ and traditional interfaces. Our observations seem to indicate that because SDAZ is a relatively easy tool to become familiar with, results will not greatly improve.

Other SDAZ applications should be explored, such as code editors and word processors. An exciting future development would be the design and evaluation of 3-D SDAZ interfaces.

Chapter 7

Conclusion

Speed-dependent automatic zooming (SDAZ) is a new technique designed to efficiently navigate through documents. It couples rate-based scrolling with automatic zooming. The idea is to automatically zoom the document as the scroll speed increases, keeping the visual flow across the screen constant. This allows users to scroll a document without having to manually switch between zooming and scrolling, and without becoming disoriented by the fast visual flow.

We have designed and implemented two SDAZ interfaces — a map browser and a document browser. Through evaluations, of SDAZ and traditional scroll and zoom interfaces, we found that SDAZ document browsing is 22% faster on average for locating pictures and text titles. SDAZ map browsing is 43% faster on average for locating targets. This data and participants' preference of both SDAZ interfaces, suggests that SDAZ has commercial applications.

Acknowledgments

The author would like to thank Andy Cockburn, Amal Sirisena, and Julian Looser for support and suggestions in the development of this work.

Bibliography

Ahlberg, C. & Shneiderman, B. (1994), The Alphaslider: A Compact and Rapid Selector, *in* 'Proceedings of CHI'94 Conference on Human Factors in Computing Systems Boston, April 24–28', ACM Press, pp. 365–371.

Barrett, R., Sleker, E., Rutledge, J. & Olyha, R. (1995), The negative inertia: A dynamic pointing function, *in* 'Conference Companion on Human factors in Computing Systems, May 07-11. Denver, Colorado, United States', ACM Press, pp. 316–317.

Beard, D. & Walker, J. (1990), 'Navigational techniques to improve the display of large two-dimensional spaces', *Behav. Inf. Tech* **9**(6), 451–466.

Bederson, B. & Hollan, J. (1994), Pad++: A zooming graphical interface for exploring alternate interface physics, *in* 'Proceedings of ACM Conference on User Interface Software and Technology, 1994'.

Cockburn, A. (2001), 'Supporting Tailorable Program Visualisation Through Literate Programming and Fisheye Views', *Information and Software Technology* **43**(13), 745–758.

Combs, T. & Bederson, B. (1999), Does zooming improve image browsing?, *in* 'Proceedings of the 4th ACM Conference on Digital libraries, Berkeley, California, United States', ACM Press, pp. 130–137.

Furnas, G. (1986), Generalized Fisheye Views, *in* 'Proceedings of the CHI'86 Conference on Human Factors in Computing Systems III', Amsterdam; North Holland/ACM, pp. 16–23.

Furnas, G. & Bederson, B. (1995), 'Space-scale diagrams: Understanding multiscale interfaces', *Proceedings of CHI 95* pp. 234–241.

Gutwin, C. (2002), Improving Focus Targeting in Interactive Fisheye Views, *in* 'Proceedings of CHI'2002 Conference on Human Factors in Computing Systems Minneapolis, Minnesota, 20–25 April', pp. 267–274.

Hart, S. & Staveland, L. (1988), Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research, *in* P. Hancock & N. Meshkati, eds, 'Human Mental Workload', Elsevier Science, pp. 139–183.

Igarashi, T. & Hinckley, K. (2000), Speed-dependent automatic zooming for browsing large documents, *in* 'Proceedings of the 13th Annual ACM symposium on User Interface Software and Technology, San Diego, California, United States', ACM Press, pp. 139–148.

Masui, T., Kashiwagi, K. & Borden IV, G. (1995), Elastic graphical interfaces for precise data manipulation, *in* 'Conference Companion on Human factors in Computing Systems, May 07-11. Denver, Colorado, United States', ACM Press, pp. 143–144.

Perlin, K. (1985), 'An image synthesizer', *Computer Graphics* **19**(3).

Perlin, K. & Fox, D. (1993), Pad: An alternative approach to the computer interface, *in* 'Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques, September.', ACM Press, pp. 57–64.

Smith, D. & Henning, R. (1996), Stretch button scrollbar, *in* 'Proceedings of CHI'96 Conference on Human Factors in Computing Systems Vancouver, April 13–18', ACM Press, pp. 301–302.

Smith, M. & Cockburn, A. (2002), 'Hidden messages: Evaluating the effectiveness of code elision in program navigation'. Submitted to the International Journal of Empirical Software Engineering.

Tan, D., Robertson, G. & Czerwinski, M. (2001), Exploring 3D Navigation: Combining Speed-coupled Flying with Orbiting, *in* 'Proceedings of CHI'2001 Conference on Human Factors in Computing Systems Seattle, Washington, March 31–April 6', pp. 418–425.

Ware, C. & Fleet, D. (1997), Context sensitive flying interface, *in* 'Proceedings of the 1997 Symposium on Interactive 3D Graphics, Providence, Rhode Island, United States', ACM Press, pp. 127–130.

Zhai, S., Smith, B. & Selker, T. (1993), Improving browsing performance: A study of four input devices for scrolling and pointing tasks, *in* 'INTERACT '93 and CHI '93 Conference Companion on Human Factors in Computing Systems. April 24-29. Amsterdam, The Netherlands', ACM Press, pp. 123–125.